

[Time for: #Smalltalk]

anders.janmyr@factor10.com

<http://anders.janmyr.com>

factor *10*



exampleWithNumber: x

"The example has Unary, binary, and key word messages, declares arguments and temporaries (but not block temporaries), accesses a global variable (but not class and instance variable), uses literals (array, character, symbol, string, integer, float), uses the pseudo variables true, false, nil, self, and super, and has sequence, assignment, return and cascade. It has both zero argument and one argument blocks. "

| y |

```
true & false not nil  
iffalse: [self hate].
```

```
y := self size + super size.
```

```
#$a #a 'a' 1 1.0 )
```

```
do: [:each | Transcript show: each class name;
```

```
show: each printString;
```

```
show: ' '].
```

```
^ x < y
```

The Language

Pseudo Variables

<code>true.</code>	"One of two instances of Boolean"
<code>false.</code>	"The other one"
<code>nil.</code>	"The single instance of UndefinedObject"
<code>self.</code>	"This"
<code>super.</code>	"Parent"
<code>thisContext.</code>	"This running stackframe"

Messages

Unary Message

```
Tapir>>
```

```
  swim
```

```
    "Tells the tapir to swim"
```

```
    Transcript show: 'Swim, swim, swim...'
```

```
| aTapir |
```

```
aTapir := Tapir new.
```

```
aTapir swim "Swim, swim, swim... in the Transcript window"
```

Binary Message

```
Tapir>>  
  + aTapir  
    "Mate with another tapir"  
    ^ self mate: aTapir.
```

```
| lisa dominic lido |  
lisa := Tapir new.  
dominic := Tapir new.  
lido := lisa + dominic
```

Keyword Message

```
Tapir>>
```

```
  sleep: aDuration in: aPlace
```

```
    "Tells the tapir to go to sleep somewhere"
```

```
    | message |
```

```
      message := 'I'm going to ', aPlace,  
                 ' to sleep for ', aDuration.
```

```
      Transcript show: message.
```

```
Tapir new sleep: '2 minutes' in: 'the jungle'
```

```
"Shows: I'm going to the jungle to sleep for 2 minutes"
```

Class Declaration

```
Object subclass: #Tapir  
  instanceVariableNames: 'name kind'  
  category: 'Animals'
```

doesNotUnderstand:

```
doesNotUnderstand: aMessage  
  | index |  
  ^(columns includesKey: aMessage selector)  
    ifTrue:  
      [index := accessTable: (aMessage selector)]  
    ifFalse:  
      [super doesNotUnderstand: aMessage].
```

[Blocks]

Demo



Conditionals

```
(17 * 13 > 220)  
  ifTrue: [ 'bigger' ]  
  ifFalse: [ 'smaller' ]
```

Conditionals

Boolean>>

```
ifTrue: trueAltBlock ifFalse: falseAltBlock  
  self subclassResponsibility
```

Conditionals

False>>

```
if True: trueAltBlock if False: falseAltBlock  
    ^falseAltBlock value
```

Conditionals

True>>

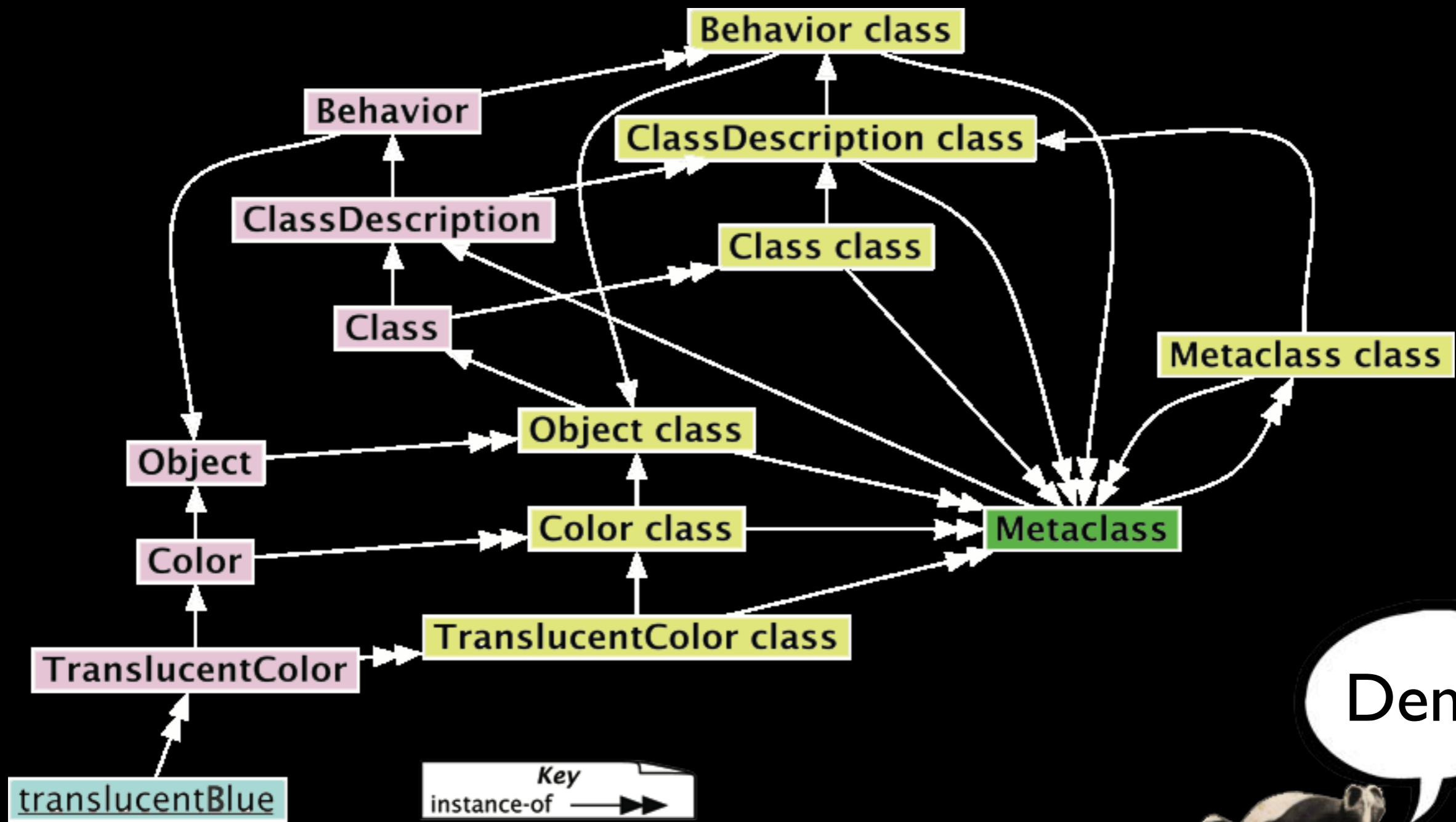
```
if True: trueAltBlock if False: falseAltBlock
    ^trueAltBlock value
```

Loops

```
[ x < 10 ] whileTrue: [  
  Transcript show: x.  
  x := x + 1 ].
```

Exceptions

```
[fileStream := FileStream named: nameString]
  on: FileNotFoundException
  do: [:ex |
    Transcript show: 'File not found: ', ex fileName.]
```



Demo



Persistence

Save Image

```
[  
  [  
    (Delay forSeconds: 300) wait.  
    SmalltalkImage current saveSession.  
  ] repeat.  
] forkAt: Processor userBackgroundPriority.
```

Object Databases

Magma

```
"Magma setup root"  
MagmaRepositoryController  
  create: 'c:\myMagmaFolder'  
  root: Dictionary new
```

Magma

```
"Magma get session"  
| mySession |  
mySession := MagmaSession  
    host: 'localhost'  
    port: 51001.  
mySession connectAs: 'anders'
```

Magma

“Magma add and commit”

mySession commit:

[mySession root

at: 'tapirs'

put: (OrderedCollection with:

(Tapir name: 'Carl-Philip'))]

Gemstone

Gemstone

```
"Gemstone query"
```

```
| mySet |
```

```
mySet := myEmployees select: { :anEmployee |  
    (anEmployee.name = 'Conan')  
    & (anEmployee.job = 'librarian')  
}
```

Gemstone

```
"Gemstone Migrations"  
method: Point  
migrateFrom: oldPoint instVarMap: aMap  
  | x y |  
  x := oldPoint x.  
  y := oldPoint y.  
  radius := ((x*x) + (y*y)) asFloat sqrt.  
  angle := (x/y) asFloat arcTan.  
  ^self
```

O-R Mappers

GLORP

"GLORP Mapping"

```
mapping := OneToOneMapping new
  attributeName: #custNo;
  referenceClass: MyCustomer;
  mappingCriteria: (PrimaryKeyExpression
    from: (table fieldNamed: 'ADDRESS_ID')
    to: (table fieldNamed: 'ID')).
```

GLORP

"Glorp Query"

aMonthAgo := Date today subtractDays: 30.

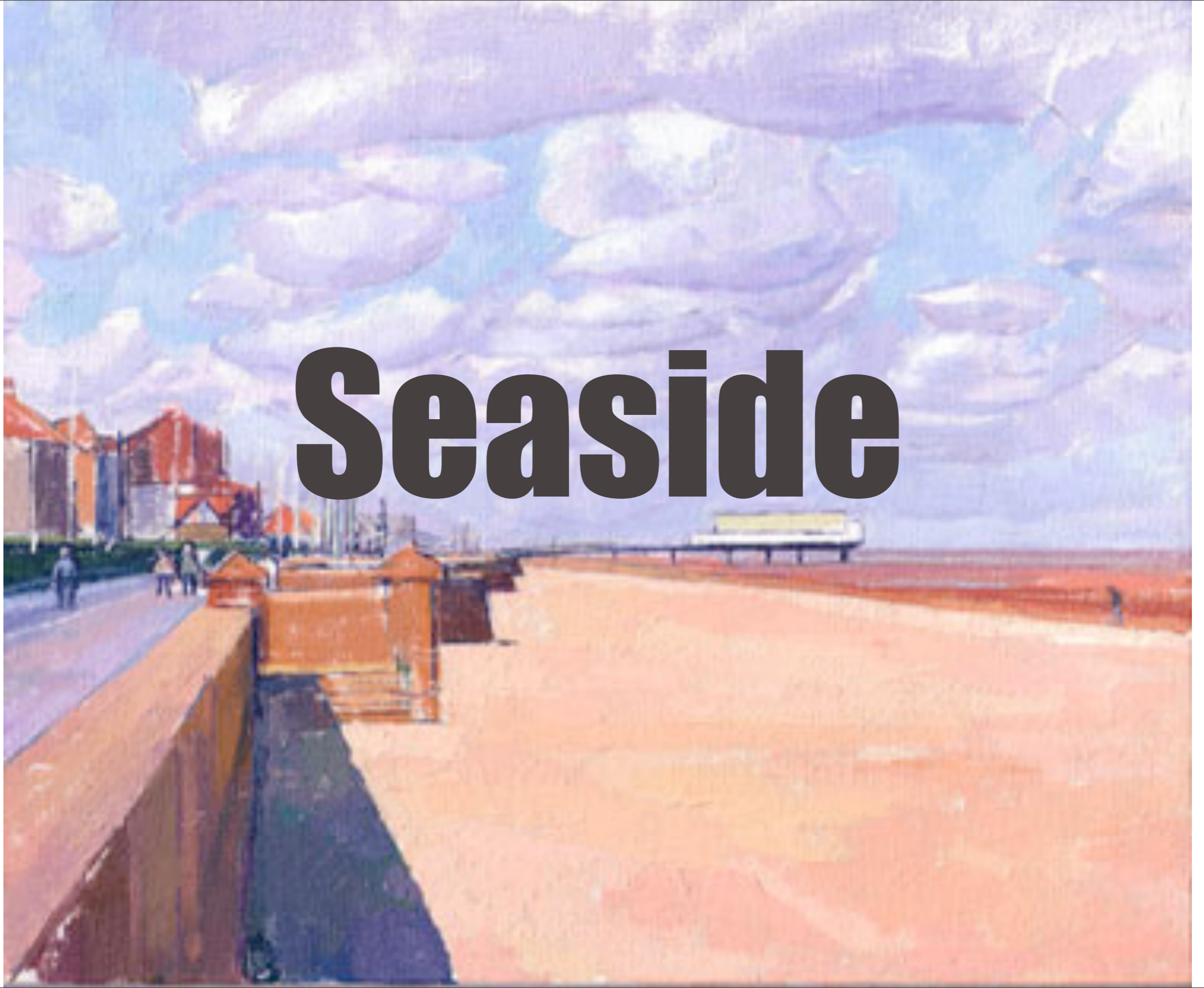
newUsers := session

 readManyOf: User

 where: [:each | each joined > aMonthAgo].

"SELECT ... FROM TUT_USER t1 WHERE t1.JOINED > ?"

Seaside



Component Based

```
PBBlogHtmlView >>
```

```
  renderContentOn: html
```

```
    super renderContentOn: html.
```

```
    self renderPosts: self batcher batch on: html.
```

```
    self renderBatcherOn: html.
```

```
    self renderRssOn: html
```

Templating in Smalltalk

```
renderContentOn: html
  html heading: count.
  html anchor
    callback: [ self increase ];
    with: '++'.
  html space.
  html anchor
    callback: [ self decrease ];
    with: '--'
```

Continuations

```
html anchor  
  callback: [ self increase ];  
  with: '++'.
```



The Image

**Smalltalk is
written in
itself!**



Alive!



Demo

**Less
is
More**

less



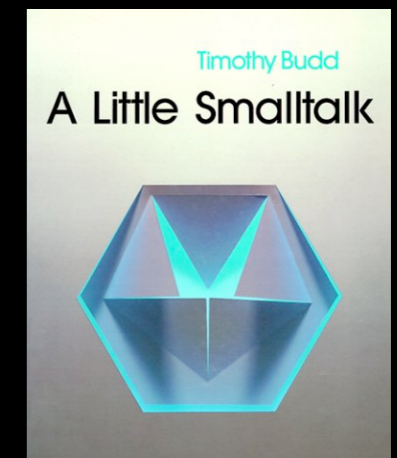
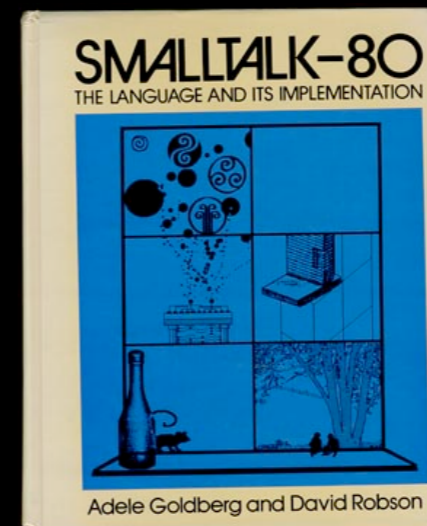
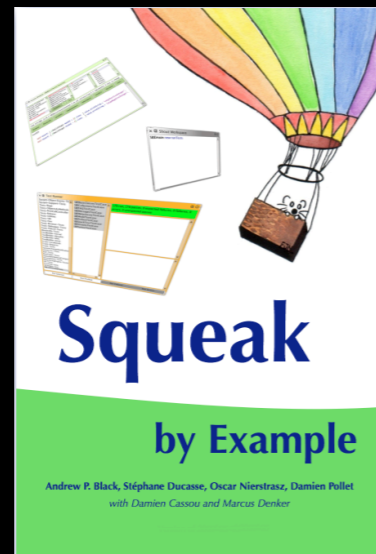
more

Less

is

Less!

www.squeak.org



Thank You!
Questions?

<http://anders.janmyr.com>